## REMARKS

Claims 1-24 are presented for further examination. Claims 1, 5, 10, 15, and 20 have been amended.

In the Office Action mailed July 7, 2004, the Examiner rejected claims 1-11 and 15-21 under 35 U.S.C. § 102(b) as anticipated by U.S. Patent No. 5,832,219 ("Pettus"). Claims 14 and 24 were rejected under 35 U.S.C. § 103(a) as unpatentable over Pettus in view of "Official Notice." Claims 12 and 22 were rejected as obvious over Pettus in view of U.S. Patent No. 6,182,068 ("Lomet et al.").

Applicants have amended claims 1, 5, 10, 15, and 20 as set forth above to recite that communication is via a message queue. In the method and system of the present invention, communication is made via a message queue or a shared memory message queue. The interface method used in the present invention is clearly different than that described by Pettus.

More particularly, Pettus describes an object-oriented RPC implementation. Requests from clients to servers in the system contain at least a class name and a request number. At the server side, a dispatcher object looks up the appropriate function to run based on this information. Specifically, the dispatcher maintains a dictionary that maps classes to offsets in a request table. The offset into the request table is added to the particular request number in order to find the function pointer that refers to the actual function to execute. Hence, Pettus focuses on providing a reference to a service object, which is distinct from a reference to an interface method as used in the present invention. Moreover, the Pettus reference relied upon by the Examiner is quite clearly directed towards client and server components communicating via a network. Nowhere does Pettus teach or suggest communication via a message queue because the client and server components all communicate via networking interfaces and not a message queue.

Claim 1 has been amended to include the additional step of "sending client component requests to server components via a message queue." Independent claims 5, 10, 15, and 20 each have been amended to recite an interface manager that communicates via a message queue, or in the case of method claim 5, receiving a request "via a message queue."

As discussed above, nowhere does Pettus teach or suggest the use of a message queue for communication with both client and server components. For these reasons, applicants respectfully submit that independent claims 1, 5, 10, 15, and 20, as well as all claims depending therefrom, are clearly allowable.

More particularly, claims 3, 7-8, and 19 are directed to *clients*, not *servers* (see Pettus, column 12, lines 2-44). Pettus at column 13, lines 11-18, describes a communication path *between* clients and servers.

Dependent claims 11 and 21 are also allowable because they are directed to a plurality of components in a network station. While the Examiner cites various references to tables of function pointers associated with servers in the Pettus reference, applicants' claims are directed to multiple components residing on one host, which Pettus does not teach or suggest. More particularly, three separate components are claimed: (1) generic server components with tables of function pointers; (2) a station manager having references to the tables of pointers and the components; and (3) an interface manager for communicating with the components of the station manager. It appears the Examiner is corresponding (1) to server components, (2) to client components, and (3) to one of dispatcher objects (see Pettus, Figure 9, which is part of the servers), clients (see Pettus at column 9, lines 49-58), or communication links (see Pettus at column 12, lines 10-20). This correspondence is indefinite because the interface manager cannot be both a server, client, and communications link all at the same time.

With respect to claims 11 and 21, the confusion from above carries forth into the rejection of these claims. It appears the Examiner is associating the station manager with the client but then associating the interface manager with the server. Under this interpretation, the server corresponds to *both* the interface manager and the plurality of components, which is not possible.

With respect to dependent claims 14 and 24, which were rejected as obvious over Pettus in view of Official Notice, the Examiner takes Official Notice of storing function pointers in a shared memory area. Pettus, however, is directed to a client-server architecture wherein clients and servers are located on different hosts. In fact, the entire point of RPC is to make remotely located services appear as if they are services located on the local machine, because this

abstraction allows client programs to interact with remote servers via ordinary, local function calls (which in turn hide the complexity of opening network connections and transporting parameters and return values over the network). Thus, Pettus can be said to teach away from the idea of having clients and servers coexisting on a local machine and the inventive optimization of placing function pointers in a shared memory region.

With respect to claims 12 and 22, which were rejected as obvious over Pettus in view of Lomet et al., Lomet et al. discuss the idea of ordering requests serially (see column 8, line 40), but the purpose is related to crash recovery for a database system. Inasmuch as Lomet et al. is directed to solving a completely different problem (reliable database transactions and rollbacks), there is no teaching or suggestion in Lomet et al. that applying the techniques therein would address the problem solved by the present invention.

With respect to claims 13 and 23, which were rejected as obvious over Pettus in view of Harchol-Balter, the Harchol-Balter reference describes a distributed server system that implements load balancing. While it does discuss the idea of processing requests in a first-come first-served order (column 11, lines 46-52), this purpose is related to a load balancing application. There is no teaching or suggestion in Harchol-Balter that applying these techniques from this reference point would address the problem solved by the present invention.

In view of the foregoing, applicants submit that all of the claims in this application are clearly allowable. In the event the Examiner finds informalities that can be resolved by telephone conference, the Examiner is urged to contact applicants' undersigned representative by telephone at (206) 622-4900 in order to expeditiously resolve prosecution of this application. Consequently, early and favorable action allowing these claims and passing this case to issuance is respectfully solicited.

The Director is authorized to charge any additional fees due by way of this Amendment, or credit any overpayment, to our Deposit Account No. 19-1090.

All of the claims remaining in the application are now clearly allowable. Favorable consideration and a Notice of Allowance are earnestly solicited.

Respectfully submitted,

SEED Intellectual Property Law Group PLLC

E. Russell Tarleton
Registration No. 31,800

ERT:jl

Enclosure:
Postcard

701 Fifth Avenue, Suite 6300
Seattle, Washington 98104-7092
Phone: (206) 622-4900
Fax: (206) 682-6031

521858_1.DOC